DATE AND TIME IN RUBY

The **Time** class represents dates and times in Ruby. It is a thin layer over the system date and time functionality provided by the operating system. This class may be unable on your system to represent dates before 1970 or after 2038.

This tutorial will make you familiar with all the most wanted concepts of date and time.

Getting Current Date and Time:

Following is the simple example to get current date and time:

```
#!/usr/bin/ruby -w

time1 = Time.new

puts "Current Time : " + time1.inspect

# Time.now is a synonym:
time2 = Time.now
puts "Current Time : " + time2.inspect
```

This will produce following result:

```
Current Time : Mon Jun 02 12:02:39 -0700 2008
Current Time : Mon Jun 02 12:02:39 -0700 2008
```

Getting components of a Date & Time:

We can use *Time* object to get various components of date and time. Following is the example showing the same:

```
#!/usr/bin/ruby -w

time = Time.new

# Components of a Time
puts "Current Time : " + time.inspect
puts time.year    # => Year of the date
puts time.month   # => Month of the date (1 to 12)
puts time.day     # => Day of the date (1 to 31 )
puts time.wday    # => 0: Day of week: 0 is Sunday
puts time.yday    # => 365: Day of year
puts time.hour    # => 23: 24-hour clock
puts time.min     # => 59
puts time.sec     # => 59
puts time.usec    # => 999999: microseconds
puts time.zone    # => "UTC": timezone name
```

This will produce following result:

```
Current Time : Mon Jun 02 12:03:08 -0700 2008
2008
6
2
1
154
12
3
8
247476
UTC
```

Time.utc, *Time.gm* and *Time.local* Functions:

These two functions can be used to format date in standard format as follows:

```
# July 8, 2008
Time.local(2008, 7, 8)
# July 8, 2008, 09:10am, local time
Time.local(2008, 7, 8, 9, 10)
# July 8, 2008, 09:10 UTC
Time.utc(2008, 7, 8, 9, 10)
# July 8, 2008, 09:10:11 GMT (same as UTC)
Time.gm(2008, 7, 8, 9, 10, 11)
```

Following is the example to get all components in an array in the following format:

```
[sec,min,hour,day,month,year,wday,yday,isdst,zone]
```

Try the following:

```
#!/usr/bin/ruby -w

time = Time.new

values = time.to_a
p values
```

This will generate following result:

```
[26, 10, 12, 2, 6, 2008, 1, 154, false, "MST"]
```

This array could be passed to *Time.utc* or *Time.local* functions to get different format of dates as follows:

```
#!/usr/bin/ruby -w

time = Time.new

values = time.to_a
puts Time.utc(*values)
```

This will generate following result:

```
Mon Jun 02 12:15:36 UTC 2008
```

Followin is the way to get time represented internally as seconds since the (platform-dependent) epoch:

```
# Returns number of seconds since epoch
time = Time.now.to_i

# Convert number of seconds into Time object.
Time.at(time)

# Returns second since epoch which includes microseconds
time = Time.now.to_f
```

Timezones and daylight savings time:

You can use a *Time* object to get all the information related to Timezones and daylight savings as follows:

```
time = Time.new

# Here is the interpretation
```

```
time.zone        # => "UTC": return the timezone
time.utc_offset # => 0: UTC is 0 seconds offset from UTC
time.zone        # => "PST" (or whatever your timezone is)
time.isdst       # => false: If UTC does not have DST.
time.utc?        # => true: if t is in UTC time zone
time.localtime  # Convert to local timezone.
time.gmtime      # Convert back to UTC.
time.getlocal   # Return a new Time object in local zone
time.getutc      # Return a new Time object in UTC
```

Formatting Times and Dates:

There are various ways to format date and time. Here is one example showing few:

```
#!/usr/bin/ruby -w
time = Time.new

puts time.to_s
puts time.ctime
puts time.localtime
puts time.strftime("%Y-%m-%d %H:%M:%S")
```

This will produce following result:

```
Mon Jun 02 12:35:19 -0700 2008
Mon Jun  2 12:35:19 2008
Mon Jun 02 12:35:19 -0700 2008
2008-06-02 12:35:19
```

Time Formatting Directives:

These directives in the following table are used with the method *Time.strftime*.

Directive	Description
%a	The abbreviated weekday name (Sun).
%A	The full weekday name (Sunday).
%b	The abbreviated month name (Jan).
%B	The full month name (January).
%c	The preferred local date and time representation.
%d	Day of the month (01 to 31).
%H	Hour of the day, 24-hour clock (00 to 23).
%I	Hour of the day, 12-hour clock (01 to 12).
%j	Day of the year (001 to 366).
%m	Month of the year (01 to 12).
%M	Minute of the hour (00 to 59).
%p	Meridian indicator (AM or PM).
%S	Second of the minute (00 to 60).

%U	Week number of the current year, starting with the first Sunday as the first day of the first week (00 to 53).
%W	Week number of the current year, starting with the first Monday as the first day of the first week (00 to 53).
%w	Day of the week (Sunday is 0, 0 to 6).
%x	Preferred representation for the date alone, no time.
%X	Preferred representation for the time alone, no date.
%y	Year without a century (00 to 99).
%Y	Year with century.
%Z	Time zone name.
%%	Literal % character.

Time arithmetic:

You can do simple arithmetic with time as follows:

```
now = Time.now      # Current time
past = now - 10     # 10 seconds ago. Time - number => Time
future = now + 10   # 10 seconds from now Time + number => Time
future - now        # => 10  Time - Time => number of seconds
```